The 13th Technology of Deep Space One

Abstract

Nicolas Rouquette & Peter Gluck
Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California, USA

On October 24th, 1998, the Deep Space One (DS-1) spacecraft launched aboard a Delta II rocket as the first step towards the bold task of testing and validating 12 new technologies for future missions. This launch also represented yet another thrilling event; namely, the successful test and validation of a 13th heretofore undisclosed technology: **model-based code-generation** of the spacecraft's system-level fault-protection (FP) software from behavioral state diagrams and structural models.

validation

Until DS-1, the Jet Propulsion Laboratory (JPL) had not used code-generation techniques on large scale for avionics software. However, the constraints of the mission design and development cycle, limited budget and resources dictated a departure from past practices. The analysis of the system-level issues started in March 1997 with minimal staff while the actual design and development of the fault-protection engine started in earnest in June 1997. Radical departures from past projects were necessary to complete the design, development and testing of the system-level fault-protection in time for launch. The requirement that post-launch activities be directed by fault protection further increased the difficulty of the task; on other spacecraft, such activities are typically handled with sequences.

First, a decision was made in June to use the successful Mars Pathfinder (MPF) fault-protection engine because this system made a nice separation of the various concerns between detecting faults, signaling faults and executing fault responses. However, the limited resources available precluded a duplication of MPF's design and development approach because we had too few software engineers and too much uncertainty about the hardware, the flight software and the scope of the system-level issues. This high degree of uncertainty translates into a high degree of design instability and volatility. To accommodate this difficult state of affairs, we shifted the design and development of the system-level fault-protection software from low-level concerns about the C language to higher-level concerns about system-level requirements, issues, strategy, and tradeoffs. To make this top-down design approach work in a team environment while retaining sufficient implementation flexibility, we standardized on using state diagrams and attribute specifications as design notations for describing the behavior and structure of fault-protection designs.

In this paper, we describe the process we used to leverage model-based code generation from state diagrams and structural specifications to better respond to the evolving requirements and scope of DS-1's system-level fault-protection design, development, test and operation. The evolution of the high-level design and the low-level changes in the flight software architecture and interfaces contributed to multiplying the number and frequency of fault-protection software releases thereby creating a multitude of software integration issues. To address the resulting software integration issues, we broadened the scope of code generation to other forms of model-based analysis techniques more traditionally associated with first-principle's reasoning about physical models. Additionally, we describe our in-flight launch and initial acquisition experience.